WHAT IS CLAIMED IS:

1. A method by which a disk-based distributed data storage system is organized for protecting historical records of stored data entities, the method comprising:

recording distinct states of an entity, corresponding to different moments of time, as separate entity versions coexisting within the distributed data storage system; and

assigning expiration times to the entity versions independently within each of a plurality of storage sites according to a shared set of rules, before which times deletion is prohibited.

2. The method of claim 1 in which the shared set of rules requires that unexpired entity versions not be changed.

3. The method of claim 1 in which the storage system is adapted for storing an unstructured-set of entities.

4. The method of claim 3 in which the unstructured set comprises more than a million entities.

5. The method of claim 3 in which the unstructured set comprises more than a billion entities.

6. The method of claim 1 in which the storage system associates an entity with an identifier chosen by the storage client.

7. The method of claim 1 in which the storage system associates an entity version with an identifier that depends on a hash of its contents.

8. The method of claim 1 in which a client of the distributed storage system defines mechanisms to organize the storage system into a hierarchical file system, with separately accessible entities playing the roles of files and directories.

9. The method of claim 1 or 8 in which expiration times of entity versions can be extended, and extension periods for different versions can be specified independently.

10. The method of claim 9 in which an expiration time is extended at the request of a client of the storage system.

11. The method of claim 1 in which information about the entity is replicated to a plurality of storage sites, with the set of sites chosen based on a hash.

12. The method of claim 1 or 8 in which entity versions can be accessed separately, without needing to access a larger aggregate first.

13. The method of claim 1 in which the plurality of storage sites are located in different cities.

14. The method of claim 1 in which no single individual is allowed physical access to all of the plurality of storage sites.

15. The method of claim 1 in which administrative mechanisms exist for overriding the deletion prohibition.

16. The method of claim 1 in which no single individual is given the authority to override the deletion prohibition at all of the plurality of storage sites.

17. The method of claim 1 in which the versions of the entity are assigned deposit times, and the version with the latest deposit time is considered current.

18. The method of claim 17 in which non-current versions are assigned expiration times.

19. The method of claim 17 in which the deposit time is specified by a client of the distributed storage system.

20. The method of claim 17 in which the deposit time is based on the time the deposit reaches a storage site.

21. The method of claim 19 in which the deposit time is constrained to agree with the actual time that the deposit reaches a storage site, to within predetermined limits.

22. The method of claim 21 in which the actual time is determined by clocks at the storage site, operating without reference to an external time standard.

23. The method of claim 21 in which the actual time is determined by clocks at the storage site, with a limit to a total correction applied per fixed period using an external time standard.

24. The method of claim 21 in which no constraint is imposed if the deposit time specified by the client is earlier than the latest deposit time of any existing version of the entity.

25. The method of claim 19 in which the entity is used to record the history of a file in a source file system, and an historical version of the file is added from a separate record of the file system's history with a deposit time that precedes the most current version of the entity.

26. The method of claim 21 in which the imposition of the constraint begins at a predefined event, before which event versions of the entity are deposited with deposit times that violate the constraint.

27. The method of claim 26 in which the predefined event is the deposit of a version of the entity with a deposit time specified that agrees with the actual time, to within predetermined limits.

28. The method of claim 26 in which the predefined event is a request from a storage client to begin monitoring deposit times for the entity.

29. The method of claim 25 or 27 in which a client of the distributed storage system deposits records of a source file system's history into the storage system, with entities corresponding to files and directories, and the deposit times specified for versions of entities correspond to times associated with the records.

30. The method of claim 29 in which two distinct entities, each of which holds records of the content of a file in the source file system during different time intervals, are linked within a third entity.

31. The method of claim 30 in which the third entity is associated with a directory in the source file system.

32. The method of claim 17 in which the expiration time assigned to a non-current version depends on when it was superseded as the current version.

33. The method of claim 32 in which the expiration time assigned to the non-current version depends on the deposit time that was assigned to it when it was current.

34. The method of claim 33 in which the expiration time assigned to the non-current version depends on the deposit time assigned to the version that superseded it as the current version.

35. The method of claim 33 in which the expiration time assigned to the non-current version depends on the actual time when it was superseded as the current version.

36. The method of claim 17 in which the storage client supplies information that allows the storage system to associate a version with the version that it supersedes as the current version.

5

37. The method of claim 36 in which the information supplied by the storage client allows the storage system to order the versions of the entity by deposit time.

38. The method of claim 34 wherein the expiration time depends on the length of the

10   time interval during which the version was current.

39. The method of claim 34 wherein the expiration time depends upon which defined snapshot moments the version was current during.

15   40. The method of claim 32 wherein the expiration time depends on the deposit times of non-current versions of the entity.

41. The method of claim 1 wherein a version is deposited, and the expiration time for it is set by the storage client.

20

42. The method of claim 1 wherein a version is deposited, and a time interval during which it is presumed to have been current is assigned by the storage client.

43. The method of claim 42 wherein the expiration time depends on the time interval

25   during which a version is presumed to have been current.

44. The method of claim 1 in which a plurality of versions of a first entity which are deposited during a time interval all have their expiration times extended to at least a first expiration time.

30

45. The method of claim 44 in which a second entity which records hierarchical directory information including that of the first entity has a version deposited during the time interval which expires earlier than the first expiration time.

5   46. The method of claim 45 in which summary information is stored in a version of the second entity that does not expire before the first expiration time, that is sufficient to recreate hierarchical directory information of the version that does.

47. The method of claim 34 in which a version makes reference to constituent blocks

10   of stored content, with each block assigned a reference count which reflects the number of references there are to the block in any version.

48. The method of claim 47 in which the version is deleted by a storage client, the reference counts assigned to its constituent blocks of stored content are decremented, and a

15   block with reference count of zero is discarded and its storage space is reused.

49. The method of claim 34 in which versions make reference to constituent blocks of stored content, with each block assigned a reference count which reflects the number of references there are to the block in current versions.

20

50. The method of claim 49 in which each block is also assigned an expiration time that depends on the latest of expiration times associated with versions which make reference to it.

25   51. The method of claim 50 in which a block which has a reference count of zero and an expiration time which has passed is discarded, and its storage space is reused.

52. The method of claim 48 or 51 in which the reference counts for blocks of stored content are incremented when the blocks are deposited.

30

53. The method of claim 51 in which the expiration time for a block of stored content is set to a default non-zero value when the block is deposited.

54. The method of claim 17 in which entities are associated with entity version records, with each entity version record storing the association between an entity identifier freely chosen by a storage client and the versions of the entity.

55. The method of claim 54 in which each entity version record is assigned a reference count which reflects the number of references there are to the corresponding entity from within current entity versions.

56. The method of claim 55 in which each entity version record is also assigned an expiration time that depends on the latest of all of the expiration times associated with the versions of the entity recorded in the version record.

57. The method of claim 56 in which an entity version record with reference count of zero and an expiration time which has passed is discarded and the storage space is reused.

58. The method of claim 57 in which the expiration time for an entity version record is set to a default non-zero value when it is created.

59. The method of claim 48 or 51 in which the blocks of stored content are strings of bytes with a predetermined maximum length.

60. The method of claim 59 in which a block is referenced using a block name which depends upon a hash of the content of the block.

61. The method of claim 60 in which the block content has been encrypted using a key derived from its unencrypted content.

62. A method by which a disk-based distributed data storage system is organized for protecting historical records of stored data entities, the method comprising:

recording distinct states of an entity, corresponding to different moments of time, as separate entity versions coexisting within the distributed data storage system;

associating time-intervals with entity versions, corresponding to the times during which each entity version was considered current;

sharing a set of rules for retaining entity versions among a plurality of storage sites; and

designating some entity versions as deletable and some as undeletable independently at each of the plurality of storage sites.

63. The method of claim 62 in which, except for deletion, entity versions are immutable.

64. The method of claim 62 in which expiration times are also assigned to some entity versions, independently within each of the plurality of storage sites, according to a shared set of rules, before which times deletion is prohibited.

65. The method of claim 62 in which no single individual is given the authority to override the deletion prohibition at all of the plurality of storage sites.

66. A method by which a disk-based data storage system is organized for protecting historical records of stored data entities, the method comprising:

recording distinct states of an entity, corresponding to different moments of time, as separate entity versions coexisting within the data storage system;

assigning expiration times to the entity versions, before which times deletion is prohibited; and

assigning expiration times to blocks of stored content that constitute the entity versions, with at least one block shared between different entities.

44

67. The method of claim 66 in which the data storage system is distributed and the expiration times are assigned independently within each of a plurality of storage sites according to a shared set of rules.

68. The method of claim 66 in which the expiration time assigned to a block reflects the latest of the expiration times associated with a plurality of versions which make reference to it.

69. The method of claim 66 in which a block is assigned a reference count which reflects the number of references there are to the block in a plurality of versions which are not scheduled to expire.

70. The method of claim 69 in which a block is assigned a reference count which reflects the number of references there are to the block in a plurality of versions which are scheduled to expire during some specified finite time period.

71. The method of claim 69 in which the block is also assigned a default expiration time that depends on a time of origin associated with the block itself.

72. The method of claim 71 in which the default expiration time depends upon the expiration times assigned to each of a plurality of versions which make reference to the block.

73. The method of claim 71 in which a block with a reference count of zero and a default expiration time which has passed is discarded and its storage space is reused.

74. The method of claim 71 in which an authorized storage client causes a block to be discarded which has a default expiration time which has not yet passed.

75. The method of claim 71 in which an authorized storage client causes a version to be deleted for which the assigned expiration time has not yet passed.

76. The method of claim 75 in which a block referenced by the deleted version is discarded and its storage space is reused.

77. A method for keeping track of when all references of a specified category made to elements have been removed, the method being designed to fail in a manner that does not falsely conclude there are no references, the method comprising:

computing a hash value that identifies the source of a reference;

combining hash values using a first operation to record the addition of references;

combining hash values using a second operation to record the removal of references; and

concluding that reference additions for an element have been matched by reference removals.

78. The method of claim 77 in which the hash is a cryptographic hash.

79. The method of claim 77 in which the first operation includes counting the number of reference additions.

80. The method of claim 77 in which the first operation includes adding together hashes.

81. The method of claim 77 in which the first operation includes adding corresponding bits of hashes together modulo 2.

82. The method of claim 77 in which the hash value uniquely identifies the reference source.

83. The method of claim 77 in which additional information not needed to identify the reference source is included in the identifying hash.

84. The method of claim 83 in which hash values are combined at a physical location that is separated from a source of references.

85. The method of claim 84 in which the additional information is examined at the location where the hash values are combined, and a decision is made to not combine a hash value.

86. The method of claim 84 in which the additional information is examined at the location where the hash values are combined, and determines which categories of combined hash will be affected.

87. The method of claim 86 in which a reference-removal operation is performed on one category of combined hash and a reference-addition operation is performed on another.

88. The method of claim 84 in which reference sources and combined hashes are distributed among a collection of computers.

89. The method of claim 88 in which the computers are servers in a disk-based data storage system.

90. The method of claim 89 in which the data storage system is organized for protecting historical records of stored data entities.

91. The method of claim 89 in which distinct states of an entity are recorded, corresponding to different moments of time, as separate entity versions coexisting within the data storage system.

92. The method of claim 91 in which expiration times are assigned to the entity versions, before which times deletion is prohibited.

93. The method of claim 92 in which expiration times are assigned according to a shared set of rules.

94. The method of claim 93 in which expiration times are assigned to blocks of stored content that constitute the entity versions.

95. The method of claim 94 in which a hash value identifies the reference of an entity version to a block that is shared with other entities.

96. The method of claim 95 in which information about the shared set of rules is included in the reference-identifying hash.

97. The method of claim 95 in which information that allows the general deletion prohibition to be ignored is included in the reference-identifying hash.

98. The method of claim 95 in which reference additions to the shared block have been matched by reference removals, and the shared block is discarded and its storage space is reused.

99. A method by which more than one client program connected to a network stores the same data item on a storage device of a data repository connected to the network, the method comprising:

    encrypting the data item using a key derived from the content of the data item;

    determining a digital fingerprint of the data item;

    storing the data item on the storage device at a location or locations associated with the digital fingerprint; and

    assigning an expiration time to the data item, before which time deletion is prohibited.

100. The method of claim 99 in which rules governing expiration and deletion are distributed among a plurality of storage sites.

101. The method of claim 99 in which the expiration time assigned to the data item depends upon expiration times assigned by the client programs.

102. A method by which more than one client program connected to a network stores the same data item on a storage device of a data repository connected to the network, the method comprising:

determining a digital fingerprint of the data item;

testing for whether the data item is already stored in the repository by comparing the digital fingerprint of the data item to the digital fingerprints of data items already in storage in the repository;

challenging a client that is attempting to deposit a data item already stored in the repository, to ascertain that the client has the full data item; and

assigning an expiration time to the data item, before which time deletion is prohibited.

103. The method of claim 102 in which rules governing expiration and deletion are distributed among a plurality of storage sites.

104. The method of claim 102 in which the expiration time assigned to the data item depends upon expiration times assigned by the client programs.

105. A method by which more than one client program connected to a network stores the same data item on a storage device of a data repository connected to the network, the method comprising:

determining a digital fingerprint of the data item;

storing the data item on the storage device at a location or locations associated with the digital fingerprint;

associating the data item with each of a plurality of access-authorization credentials, each of which is uniquely associated with an access owner;

assigning an expiration time to the data item, before which time deletion is

prohibited; and

preparing a digital time stamp of a plurality of records associating data-items and

credentials, to allow a property of these records to be proven at a later date.

5

106. The method of claim 105 in which rules governing expiration and deletion are

distributed among a plurality of storage sites.

107. The method of claim 105 in which the expiration time assigned to the data item

10   depends upon expiration times assigned by the client programs.

108. A method by which more than one client connected to a network stores the same

data item on a storage device of a data repository connected to the network, the method

comprising:

15   determining a digital fingerprint of the data item;

testing for whether a data item is already stored in the repository by comparing the

digital fingerprint of the data item to the digital fingerprints of data items already in storage

in the repository;

associating with a data item an informational tag which may be read by at least some

20   client programs; and

assigning an expiration time to the tagged data item, before which time deletion is

prohibited.

109. The method of claim 108 in which rules governing expiration and deletion are

25   distributed among a plurality of storage sites.

110. The method of claim 108 in which the expiration time assigned to the tagged

data item depends upon expiration times assigned by the client programs.

30   111. A method by which a client connected to a data repository over a lower speed

network connection may provide higher speed access to a data item for application

processing than is possible over the relatively low speed connection to the network, the method comprising:

determining a digital fingerprint of the data item;

testing for whether the data item is already stored in a repository by comparing the digital fingerprint of the data item to digital fingerprints of data items already in the repository;

only if the data item is not already in the repository, transferring the data item over the lower speed connection from the client to the repository;

assigning an expiration time to the data item, before which time deletion is prohibited;

making a higher speed connection between an application server and the data repository;

executing an application on the application server to process the data item stored on the data repository; and

returning at least some of the processed data to the client across the lower speed connection.

112. The method of claim 111 in which rules governing expiration and deletion are distributed among a plurality of storage sites.

113. The method of claim 111 in which the expiration time assigned to the data item depends upon expiration times assigned by the client programs.

114. A method by which multiple clients browse content on a network such as the Internet, the method comprising:

each of the multiple clients accessing content on the network via one or more proxy servers;

determining the digital fingerprint of an item of content passing through the proxy server;

storing the item of content in a content repository connected to the proxy server at a location associated with the digital fingerprint;

assigning an expiration time to the item of content, before which time deletion is

prohibited;

testing for whether a content data item is already stored in the repository by

comparing the digital fingerprint of the content data item to the digital fingerprints of content

5   data items already in storage in the repository; and

associating a content data item already stored in the repository with an access

authorization credential uniquely associated with an access owner.

115. The method of claim 114 in which rules governing expiration and deletion are

10   distributed among a plurality of storage sites.

116. The method of claim 114 in which the expiration time assigned to the item of

content depends upon expiration times assigned by the multiple clients.

15   117. A method by which clients store content items which are broken into up into

smaller data items in a data repository connected to the network, the method comprising:

determining a digital fingerprint of a data item;

testing for whether a data item is already stored in the repository by comparing the

digital fingerprint of the data item to the digital fingerprints of data items already in storage

20   in the repository; and

assigning an expiration time to a data item, before which time deletion is prohibited.

118. The method of claim 117 in which rules governing expiration and deletion are

distributed among a plurality of storage sites.

25

119. The method of claim 117 in which the expiration time assigned to the data item

depends upon expiration times assigned by the multiple clients.

120. The method of claim 117 in which the expiration times assigned to data items

30   that comprise a content item depend upon an expiration time assigned to the content item.

121. The method of claim 117 in which the content item is broken up in a manner that is independent of the content.

122. The method of claim 117 in which the content item is broken up in a manner that depends on the content type.

123. The method of claim 117 in which the content item is broken up at boundaries defined by predetermined byte strings.

124. The method of claim 123 in which the choice of which byte strings constitute boundaries depends upon the value of a hash function acting on the byte strings.

125. A method for ensuring that rules that prevent premature deletion of entity versions are enforced by correctly operating servers that store the blocks of content that comprise the entity versions, the method comprising:

computing a hash value that identifies the source of a reference to a block of content;

incorporating into the hash value a description of rules or parameters that are needed in order to enforce rules; and

communicating information which allows the hash value to be computed, to a server that stores the block of content.

126. The method of claim 125 in which the hash is a cryptographic hash.

127. The method of claim 125 in which the hash value uniquely identifies the reference source.

128. The method of claim 125 in which a block of content is identified by a digital fingerprint that involves a hash of its content.

129. The method of claim 125 in which a block of content is assigned an expiration time, before which time deletion is prohibited.

130. The method of claim 125 in which the blocks of content are distributed among a plurality of storage sites.

131. The method of claim 125 in which an expiration time assigned to an entity version is also assigned to each of its constituent blocks of content.

132. The method of claim 125 in which the information which allows the hash value to be computed is included in a request to delete the block of stored content.

133. The method of claim 132 in which a server storing the block of content denies a request that violates a rule or parameter specified in the information supplied when the block was created.

134. The method of claim 125 in which distinct states of an entity are recorded, corresponding to different moments of time, as separate entity versions coexisting within a data storage system.

135. The method of claim 134 in which the rules governing deletion of an entity version depend upon when the entity version was created.

136. The method of claim 125 in which hash values that identify references to blocks of stored content are combined as part of a reference counting scheme.

137. The method of claim 136 in which some reference counts are associated with expiration times, and their values are ignored after some point in time.

138. The method of claim 1 or 125 in which the connection between an entity version and a constituent block of content is not visible to a server storing the block of content.

139. The method of claim 138 in which the stored block of content expires and the server storing it discards it and reuses its storage space.

140. The method of claim 36 in which the information supplied by the storage client that associates a version with a superseded version is discarded while the two versions are retained.

141. A method by which a distributed disk-based data storage system is organized for protecting historical records of stored data entities, the method comprising:

recording distinct states of an entity, corresponding to different moments of time, as separate entity versions coexisting within the data storage system;

assigning expiration times to the entity versions, before which times deletion is prohibited;

assigning expiration times to blocks of stored content that constitute the entity versions; and

assigning a reference count to a block of stored content that reflects the number of references there are to the block in entity versions which are scheduled to expire during some specified finite time period.

142. The method of claim 141 in which the block is also assigned a reference count that reflects the number of references there are to the block which are not scheduled to expire.

143. The method of claim 141 in which the block is also assigned a default expiration time which sets an earliest time that the block can expire, even if all expiration related reference counts are zero.

144. The method of claim 141 in which the data storage system is distributed and the expiration times are assigned independently within each of a plurality of storage sites according to a shared set of rules.

145. The method of claim 141 in which an authorized storage client causes a block to be discarded and its space reused when its expiration time has not yet passed.

146. The method of claim 141 in which an authorized storage client overrides the deletion prohibition and causes an entity version to be deleted when its expiration time has not yet passed.

147. The method of claim 146 in which a block of stored content referenced by the deleted version is discarded and its storage space is reused.

148. A method by which a disk-based data storage system is organized for protecting historical records of stored data entities, the method comprising:

recording distinct states of an entity, corresponding to different moments of time, as separate entity versions coexisting within the data storage system; and

assigning finite expiration times to entity versions based on information supplied by the storage client, before which times deletion is prohibited and after which times deletion is allowed.

149. The method of claim 148 in which a version is deposited, and the expiration time for it is set by the storage client.

150. The method of claim 148 in which a version is deposited, and a time interval during which it is presumed to have been current is assigned by the storage client.

151. The method of claim 150 in which the expiration time is assigned by a storage server and depends on the time interval during which a version is presumed to have been current.

152. The method of claim 150 in which the entity is used to record the history of a file in a source file system, and an historical version of the file is added from a separate record of the file system's history.

153. The method of claim 152 in which the added historical version has an interval during which it is presumed to be current specified that predates that of an existing version of the entity.

5

154. The method of claim 148 in which expiration times of entity versions can be extended, and extension periods for different versions can be specified independently.

155. The method of claim 148 in which unexpired entity versions cannot be changed.

10

156. The method of claim 148 in which the storage system is adapted for storing an unstructured-set of entities

157. The method of claim 148 in which the connection between an entity version and

15   a constituent block of content is not visible to a server storing the block of content.

158. The method of claim 148 in which a plurality of versions of a first entity which are deposited during a time interval all have their expiration times extended to at least a first expiration time.

20

159. The method of claim 158 in which a second entity which records hierarchical directory information including that of the first entity has a version deposited during the time interval which expires earlier than the first expiration time.

25   160. The method of claim 159 in which summary information is stored in a version of the second entity that does not expire before the first expiration time, that is sufficient to recreate hierarchical directory information of the version that does.

161. The method of claim 148 in which versions make reference to constituent blocks

30   of stored content, with each block assigned a reference count.

162. The method of claim 161 in which each block is also assigned an expiration time that depends on the latest of expiration times associated with versions which make reference to it.

5   163. The method of claim 162 in which a block which has a reference count of zero and an expiration time which has passed is discarded, and its storage space is reused.